change in data A. Data B may be an input to a function that calculates data A based on data B, for example.

[0037] In some embodiments, the object detector **202** comprises a cascade classifier that has been trained to detect non-luminous objects in the obtained images. The cascade classifier may be trained with images of non-luminous objects that a user is expected to attempt to use a video games controller. In some cases, it may be that a user is told in advance what objects can be used as a games controller, and these objects may correspond to the objects that the cascade classifier has been trained on. The cascade identifier may be configured to identify the pixels in each obtained image that correspond to the non-luminous object being held by the user. It will be appreciated that any suitably trained machine learning model may be used to detect the non-luminous object in the obtained images. Examples of machine learning models that may be trained for such purposes include Region-Based Convolutional Neural Networks (R-CNNs), Fast R-CNNs, You Only Look Once models (YOLO), etc.

[0038] In additional or alternative embodiments, the object detector **202** is configured to detect a contour in the obtained images corresponding to a periphery of the non-luminous object that is being held by the user. The 'findContours( )' function available in OpenCV® may be used for finding such contours in the obtained image. The 'findContours( )' function uses Green's theorem, followed by second and third order image moments to find the contours in a given image. Techniques for identifying contours in images are well known in the art and any suitable technique may employed by the object detector **202**.

[0039] In examples where contour detection is used to detect the object in the image, the object detector **202** may be configured to obtain colour information indicating a pre-determined or learned colour of the non-luminous object that the user is holding or intends to hold and to filter out one or more colours not corresponding to the pre-determined colour from the obtained images, prior to performing the contour detection. For example, it may be known in advance that a user intends to use a banana or orange as a video games controller (e.g. by having selected an option such as 'add a banana for player 2 to join'), and so the object detector **202** may be configured to segment out colours not corresponding to yellow or orange from the obtained images. The object detector **202** may be configured to generate a binary mask of each obtained image, following the filtering of any colours known not to correspond to the object being held by the user. The contour detection may then be performed on the binary mask generated for each image.

[0040] The object detector **202** may be configured to detect a plurality of contours in the obtained images. For example, it may be that, following a filtering operation, there are a plurality of closed loops in the obtained images (or filtered images), with only some (or one) of these corresponding to the object being held by the user. The object detector **202** may be configured to determine which of the detected contours has the largest area and to identify the largest contour as corresponding to the object being held by the user. It may be sufficient to identify the largest contour as corresponding to the object being held by the user when the user is positioned relatively close to the camera capturing the images, since further objects will generally be associated with smaller contours. Moreover, if the colour filtering operation has been performed, the largest contour may be a reliable indication of the object that is being held by the user.

[0041] In some examples, where e.g. a user's hand is positioned over some of the object, the object detector **202** may be configured to identify two contours having the largest area as corresponding to the object being held by the user. This is because the user's hand may interrupt the larger closed loop that would otherwise be formed by the object in absence of the user's hand. As will be appreciated, in some examples, it may be necessary to determine whether the two contours having the largest area are separated by more than a threshold distance, since if they are, this may be indicative that the two contours correspond to two completely different objects, one of which is not being held by the user.

[0042] As mentioned previously, the system **200** also comprises an object pose detector **203** operable to detect changes in pose of the non-luminous object based on the obtained images of the object. The object pose detector **203** is configured to detect the pose of the passive non-luminous object in the obtained images based on the obtained images of the object. The object pose detector **203** is configured to detect the pose of the passive non-luminous object based on at least one of (i) a contour detection operation and (ii) the output of a machine learning model that has been trained to detect the poses of passive non-luminous objects in images. In the embodiments described herein, the two forms of a pose detection are generally described separately, although it will be appreciated that these may be used in combination. For example the object detector may use contour detection to detect the object in an obtained image, with the detected contour being input to a suitably trained machine learning model.

[0043] In some examples, it may be that both the object detection and/or object pose detection is performed exclusively using contour detection or a suitably trained machine learning model.

[0044] In embodiments involving contour detection, the pose detector **203** may be configured to detect changes in pose of the non-luminous object based on changes in at least one of the orientation, position and area of the contour in the obtained images, relative to a default orientation, position and area of the contour. For example, the user may hold the banana in the pose that they intend to use it for e.g. 1 s, with changes in orientation, position and area of the contour being detected relative to this initial pose.

[0045] In these embodiments (i.e. involving contour detection), the object pose detector **203** may be configured to generate a linear representation of the object. For example, if the object corresponds to a banana, the object pose detector **203** may be configured to fit a line to the contour corresponding to a line along the length of the banana. This line may be fitted by taking e.g. the average (x,y) coordinate of the contour for each x-value of the contour having two y-values. The object pose detector **203** may then be configured to detect changes in orientation of the object based on a rotation of the corresponding linear representation of the object, relative to a default orientation of the linear representation of the object.

[0046] It will be appreciated that a linear representation of the contour need not be generated in order to detect changes in an area and position of the contour, although the linear representation may still be used for this purpose.